Stable Diffusion 介绍

Stable Diffusion 是一种潜在的文本到图像扩散模型，能够在给定任何文本输入的情况下生成照片般逼真的图像，培养自主自由以产生令人难以置信的图像。

https://civitai.com/ 三方流行模型下载

https://huggingface.co/ 机器学习官网，详细文档、模型资源等

生成图片的关键在于模型，模型训练是必须掌握的知识，以下是当前几种模型 ,当前版本v0.16.0：

- Unconditional Training 与文本或图像到图像模型不同，无条件图像生成不以任何文本或图像为条件。它仅生成类似于其训练数据分布的图像。
- Text-to-Image Training 文本到图像模型（如稳定扩散）从文本提示生成图像。
- Text Inversion 文本反转是一种从少量示例图像中捕获新概念的技术。虽然该技术最初是用潜在扩散模型演示的，但此后已应用于其他模型变体，如稳定扩散。学习的概念可用于更好地控制从文本到图像管道生成的图像。它在文本编码器的嵌入空间中学习新的“单词”，这些单词在文本提示中用于个性化图像生成。
- Dreambooth 是一种个性化文本到图像模型的方法，例如稳定扩散，只需给定几 （3-5） 张主题图像。它允许模型在不同的场景、姿势和视图中生成主体的上下文图像。
- LoRA Support 可在消耗更少内存的同时加速大型模型的训练。它将秩分解权重矩阵（称为**更新矩阵**）对添加到现有权重，**并且仅**训练那些新添加的权重。
- ControlNet 将条件控制添加到文本到图像扩散模型
- instructpix2pix 微调文本条件扩散模型的方法，以便它们可以遵循输入图像的编辑指令
- Custom Diffusion 自定义扩散，比如你指定狗和指定的背景，组合成新的画面。

另外 强列建议大家，不要阅读第三方文章以及不要使用第三方工具生成模型，因为它只会误导你，每个人开发的组件都有依赖，为了解决一个问题可能会引起更多的问题，所以只建议使用官方工具。

因为最终所有的三方工具都要根据它的基础开发的。

训练模型说明

Unconditional Training 在已有的训练集进行微调，平常用不到，如图1所示：

例如，要对牛津花卉数据集进行微调：

```
accelerate launch train_unconditional.py \
  --dataset_name="huggan/flowers-102-categories" \
  --resolution=64 \
  --output_dir="ddpm-ema-flowers-64" \
  --train_batch_size=16 \
  --num_epochs=100 \
  --gradient_accumulation_steps=1 \
  --learning_rate=1e-4 \
  --lr_warmup_steps=500 \
  --mixed_precision=no \
  --push_to_hub
```

图1

Text Inversion 文本反转是一种从少量示例图像中捕获新概念的技术。虽然该技术最初是用潜在扩散模型演示的，但此后已应用于其他模型变体，如稳定扩散。学习的概念可用于更好地控制从文本到图像管道生成的图像。它在文本编码器的嵌入空间中学习新的“单词”，这些单词在文本提示中用于个性化图像生成。只需使用 3-5 张图像，你就可以向模型教授新概念。如图2

训练脚本

```
export MODEL_NAME="runwayml/stable-diffusion-v1-5"
export DATA_DIR="./cat"

accelerate launch textual_inversion.py \
--pretrained_model_name_or_path=$MODEL_NAME \
--train_data_dir=$DATA_DIR \
--learnable_property="object" \
--placeholder_token="<cat-toy>" --initializer_token="toy" \
--resolution=512 \
--train_batch_size=1 \
--gradient_accumulation_steps=4 \
--max_train_steps=3000 \
--learning_rate=5.0e-04 --scale_lr \
--lr_scheduler="constant" \
--lr_warmup_steps=0 \
--output_dir="textual_inversion_cat"
```

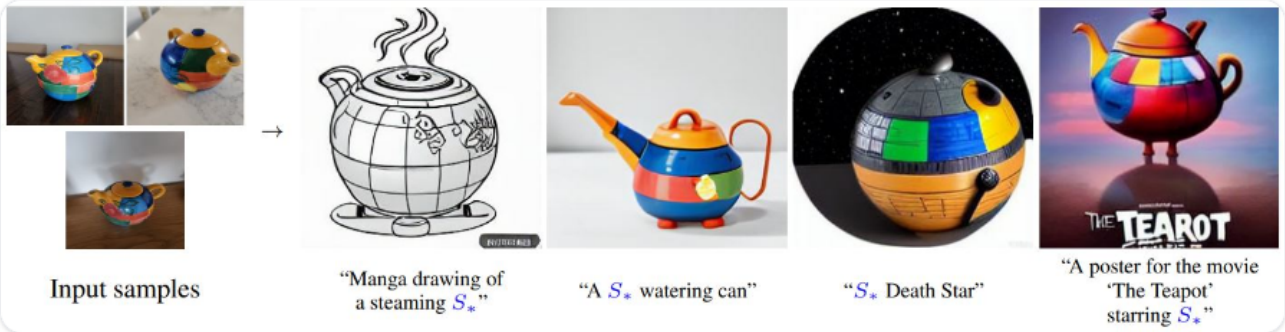


图2

DreamBooth 是一种个性化文本到图像模型的方法，例如稳定扩散，只需给定几 （3-5） 张主题图像。它允许模型在不同的场景、姿势和视图中生成主体的上下文图像。如图3

训练脚本

```
export MODEL_NAME="CompVis/stable-diffusion-v1-4"
export INSTANCE_DIR="./dog"
export OUTPUT_DIR="path_to_saved_model"

accelerate launch train_dreambooth.py \
--pretrained_model_name_or_path=$MODEL_NAME \
--instance_data_dir=$INSTANCE_DIR \
--output_dir=$OUTPUT_DIR \
--instance_prompt="a photo of sks dog" \
--resolution=512 \
--train_batch_size=1 \
--gradient_accumulation_steps=1 \
--learning_rate=5e-6 \
--lr_scheduler="constant" \
--lr_warmup_steps=0 \
--max_train_steps=400
```

图3

Text-to-image 文本到图像微调脚本是实验性的。很容易过度拟合并遇到灾难性遗忘等问题。建议探索不同的超参数，以便在数据集上获得最佳结果。文本到图像模型（如稳定扩散）从文本提示生成图像。

这种训练方式需要手动打标 。支持两种格式csv和jsonl

训练脚本

```
export MODEL_NAME="CompVis/stable-diffusion-v1-4"
export dataset_name="lambdalabs/pokemon-blip-captions"

accelerate launch --mixed_precision="fp16" train_text_to_image.py \
--pretrained_model_name_or_path=$MODEL_NAME \
--dataset_name=$dataset_name \
--use_ema \
--resolution=512 --center_crop --random_flip \
--train_batch_size=1 \
--gradient_accumulation_steps=4 \
--gradient_checkpointing \
--max_train_steps=15000 \
--learning_rate=1e-05 \
--max_grad_norm=1 \
--lr_scheduler="constant" --lr_warmup_steps=0 \
--output_dir="sd-pokemon-model"
```

图像字幕数据集具有描述图像的文本。示例可能如下所示：metadata.csv

```
file_name,text
0001.png,This is a golden retriever playing with a ball
0002.png,A german shepherd
0003.png,One chihuahua
```

或使用： `metadata.jsonl`

{"file_name": "0001.png", "additional_feature": "This is a first value of a text feature you added to your images"}
{"file_name": "0002.png", "additional_feature": "This is a second value of a text feature you added to your images"}
{"file_name": "0003.png", "additional_feature": "This is a third value of a text feature you added to your images"}

如图4案例

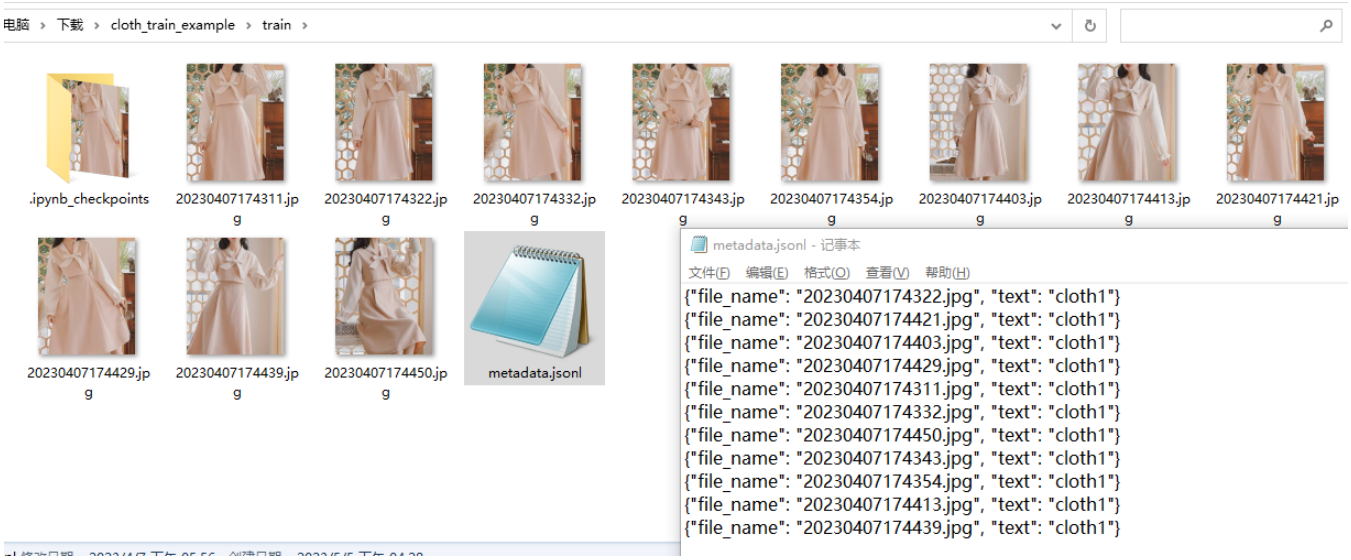


图4

Lora 模型是一种训练方法，可在消耗更少内存的同时加速大型模型的训练。它将秩分解权重矩阵（称为**更新矩阵**）对添加到现有权重，**并且仅**训练那些新添加的权重。

这有几个优点：

- 先前的预训练权重保持冻结，因此模型不容易发生灾难性遗忘。
- 秩分解矩阵的参数明显少于原始模型， 这意味着经过训练的 LoRA 权重易于移植.
- LoRA 矩阵通常添加到原始模型的注意力层中. 扩散器提供了 load_attn_procs() 方法将 LoRA 权重加载到模型的注意力层中.您可以通过参数控制模型适应新训练图像的程度。`scale`
- 更高的内存效率使您可以在特斯拉T4，RTX 3080甚至RTX 2080 Ti等消费级GPU上运行微调！像T4这样的GPU是免费的，可以在Kaggle或Google Colab笔记本中轻松访问。

lora 支持dreambooth的微调和Text-to-image微调，加载方式有两种，一种在prompt加载，一种在插件里加载

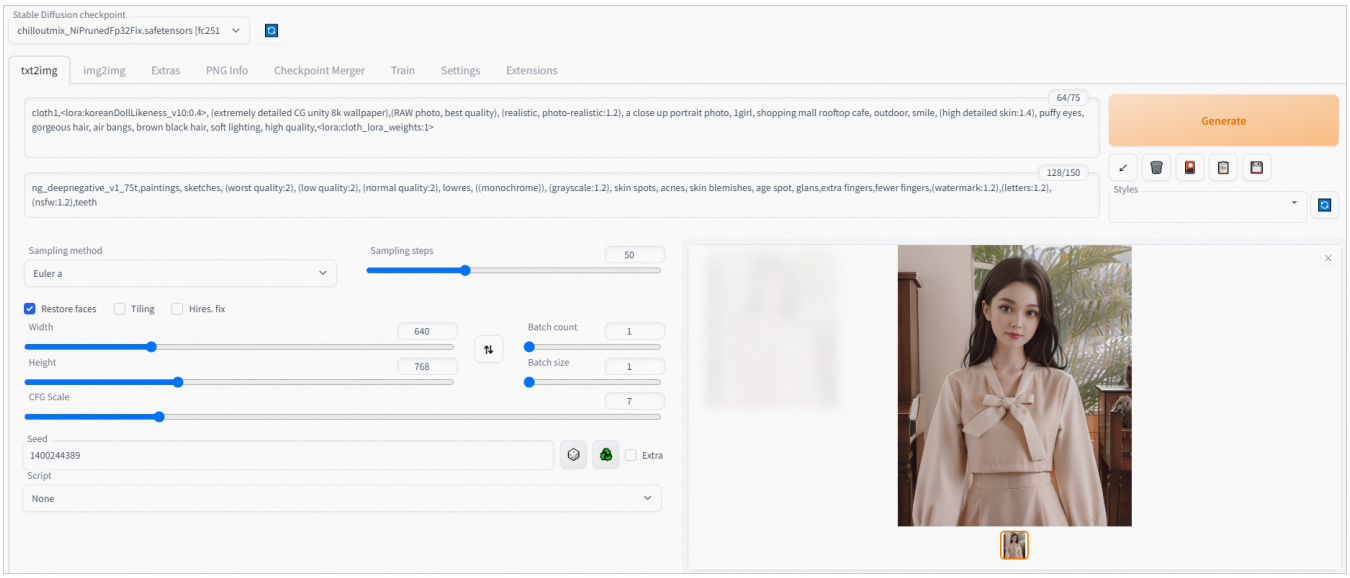
text风格lora

```
accelerate launch --mixed_precision="fp16" train_text_to_image_lora.py \
--pretrained_model_name_or_path=$MODEL_NAME \
--dataset_name=$DATASET_NAME \
--dataloader_num_workers=8 \
--resolution=512 --center_crop --random_flip \
--train_batch_size=1 \
--gradient_accumulation_steps=4 \
--max_train_steps=15000 \
--learning_rate=1e-04 \
--max_grad_norm=1 \
--lr_scheduler="cosine" --lr_warmup_steps=0 \
--output_dir=${OUTPUT_DIR} \
--push_to_hub \
--hub_model_id=${HUB_MODEL_ID} \
--report_to=wandb \
--checkpointing_steps=500 \
--validation_prompt="A pokemon with blue eyes." \
--seed=1337
```

dreambooth风格lora

```
accelerate launch train_dreambooth_lora.py \
--pretrained_model_name_or_path=$MODEL_NAME \
--instance_data_dir=$INSTANCE_DIR \
--output_dir=$OUTPUT_DIR \
--instance_prompt="a photo of sks dog" \
--resolution=512 \
--train_batch_size=1 \
--gradient_accumulation_steps=1 \
--checkpointing_steps=100 \
--learning_rate=1e-4 \
--report_to="wandb" \
--lr_scheduler="constant" \
--lr_warmup_steps=0 \
--max_train_steps=500 \
--validation_prompt="A photo of sks dog in a bucket" \
--validation_epochs=50 \
--seed="0" \
--push_to_hub
```

用人物模型给她穿上指定的衣服如下：



ControlNet 将条件控制添加到文本到图像扩散模型，可以更加精细控制某一个区域，比如只改动手部，不需要训练此模型，通用模型下载使用即可，所以不贴训练方法了。

InstructPix2Pix 是一种微调文本条件扩散模型的方法，以便它们可以遵循输入图像的编辑指令。使用此方法微调的模型采用以下内容作为输入同上不贴训练方法，如图5

个“编辑”图像，反映了应用于输入图像的编辑指令：

图5

custom_diffusion 是一种自定义文本到图像模型的方法，例如仅给定主题的几（4~5）张图像的稳定扩散。 该脚本显示了如何实现训练过程并对其进行调整以实现稳定的扩散。如图6

训练脚本

```
export MODEL_NAME="CompVis/stable-diffusion-v1-4"
export OUTPUT_DIR="path-to-save-model"
export INSTANCE_DIR="./data/cat"

accelerate launch train_custom_diffusion.py \
--pretrained_model_name_or_path=$MODEL_NAME \
--instance_data_dir=$INSTANCE_DIR \
--output_dir=$OUTPUT_DIR \
--class_data_dir=./real_reg/samples_cat/ \
--with_prior_preservation --real_prior --prior_loss_weight=1.0 \
--class_prompt="cat" --num_class_images=200 \
--instance_prompt="photo of a <new1> cat" \
--resolution=512 \
--train_batch_size=2 \
--learning_rate=1e-5 \
--lr_warmup_steps=0 \
--max_train_steps=250 \
--scale_lr --hflip \
--modifier_token "<new1>"
```

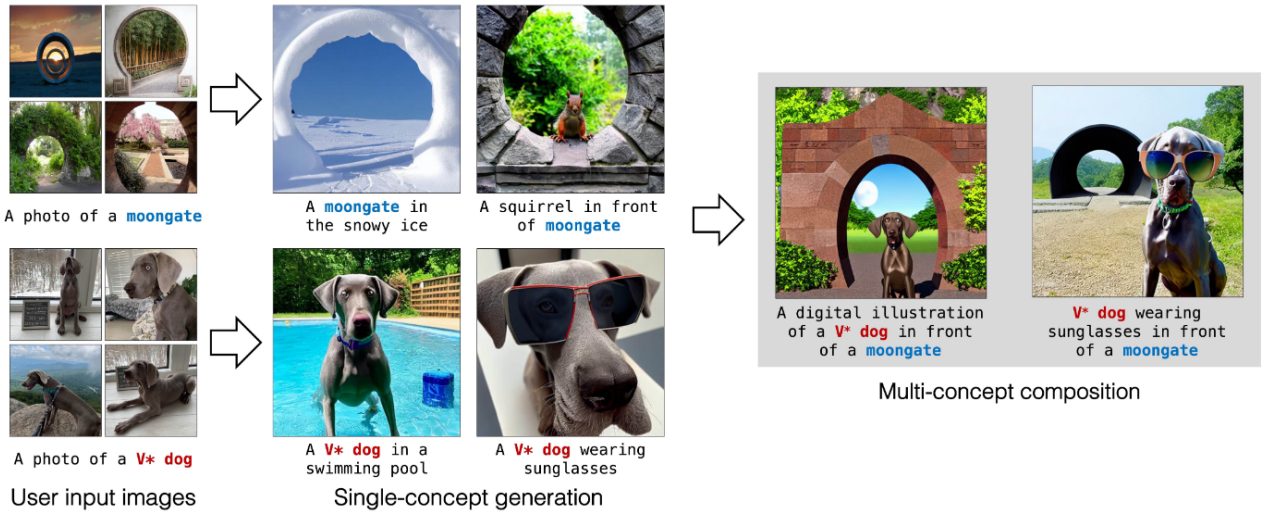


图6